Universidade Federal de Viçosa – *Campus* Florestal
Disciplina: Algoritmos e Estruturas de Dados 1 (CCF 211)
Professora: Thais Regina
Alunos: Elida Emelly – 3012, Estela Miranda – 3305,
João Marcos - 3506

Trabalho Prático 2

Complexidade de Algoritmo

Florestal / MG
2018

SUMÁRIO

1. INTRODUÇÃO

Este Trabalho Prático apresentado a Disciplina de Algoritmos e Estruturas de Dados (CCF 211) tem como objetivo fazer a Análise de Complexidade de Algoritmos aprendidos em sala de aula.

A proposta do trabalho era desenvolver um sistema, mesmo que de forma simplificada, do Problema do Caixeiro Viajante utilizado na área de Metaheurísticas, em que utilizamos de um algoritmo de permutação para olhar todas as rodas possíveis de uma cidade a outra, e retornando ao seu destino inicial.

2. CÓDIGO DE PERMUTAÇÃO

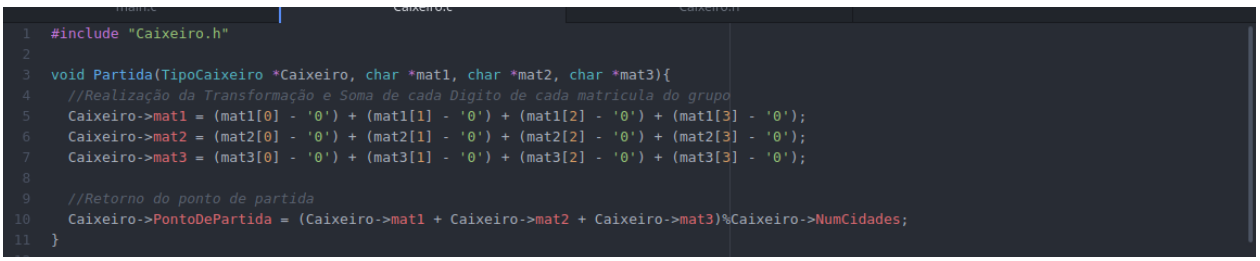
O código implementado de permutação foi retirado do site:

https://gist.github.com/marcoscastro/60f8f82298212e267021

O código base foi utilizado para encontrar as rotas possíveis, fazendo da alteração de que durante execução é feita a busca e armazenamento da rota da menor possível com base nas permutações que iniciam no ponto de partida encontrado a partir das matrículas dos alunos.

```c
void Troca(int vetor[], int i, int j){
  int aux = vetor[i];
  vetor[i] = vetor[j];
  vetor[j] = aux;
}

void Permuta(int vetor[], int inf, int sup, TipoCaixeiro Caixeiro, int cidades, int M[cidades][cidades], int *menor, int menorpermuta[cidades]){
  int soma=0;

  if(inf == sup){
    if(vetor[0] == Caixeiro.PontoDePartida){
      for(int i = 0; i <= sup+1; i++)
        printf("%d ", vetor[i]);

        for(int i=0; i<cidades; i++){
          soma += M[vetor[i]][vetor[i+1]];
        }
        printf("-> %d\n", soma);

        if(soma <= *menor){
          *menor = soma;

          for(int i=0; i<cidades; i++){
            menorpermuta[i] = vetor[i];
          }
        }
      }
  } else {
    for(int i = inf; i <= sup; i++){
      Troca(vetor, inf, i);
      Permuta(vetor, inf + 1, sup, Caixeiro, cidades, M, menor, menorpermuta);
      Troca(vetor, inf, i);
    }
    vetor[sup+1] = Caixeiro.PontoDePartida;
  }
```

Temos que para auxiliar na função de permutação utilizamos a soma, e procura do da menor distância durante a permutação (para acelerar o processamento), e uma função que realiza a procura do ponto de partida com base na matrículas dos alunos, fazendo a soma dos dígitos das mesmas.



```c
#include "Caixeiro.h"

void Partida(TipoCaixeiro *Caixeiro, char *mat1, char *mat2, char *mat3){
  //Realização da Transformação e Soma de cada Digito de cada matricula do grupo
  Caixeiro->mat1 = (mat1[0] - '0') + (mat1[1] - '0') + (mat1[2] - '0') + (mat1[3] - '0');
  Caixeiro->mat2 = (mat2[0] - '0') + (mat2[1] - '0') + (mat2[2] - '0') + (mat2[3] - '0');
  Caixeiro->mat3 = (mat3[0] - '0') + (mat3[1] - '0') + (mat3[2] - '0') + (mat3[3] - '0');

  //Retorno do ponto de partida
  Caixeiro->PontoDePartida = (Caixeiro->mat1 + Caixeiro->mat2 + Caixeiro->mat3)%Caixeiro->NumCidades;
}
```

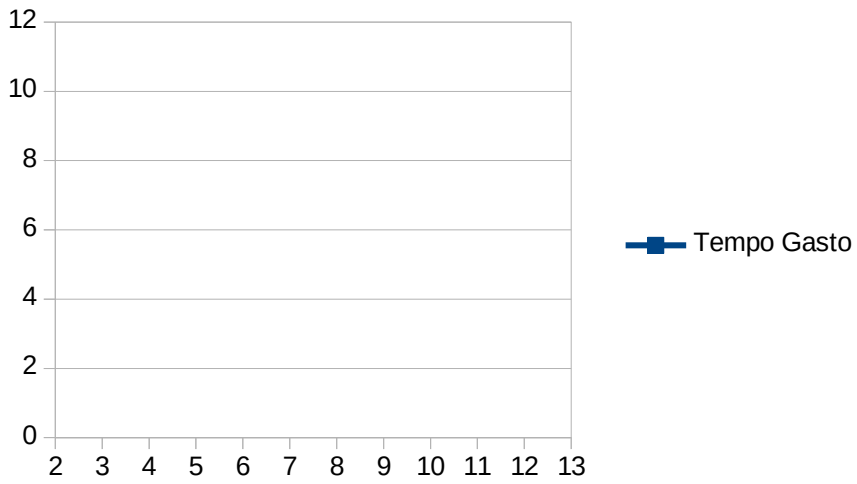
3. RESULTADOS DE EXECUÇÃO

Configurações do Computador utilizado nos testes:

Processador	Intel Core i3 2.53GHz
Memória RAM	6GB
Armazenamento Interno	500GB

Nos casos de teste foram obtidos em Segundos os seguintes resultados com base no número de cidades:

Número de Cidades	Tempo Gasto
2	0.000500s
3	0.000548s
4	0.000451s
5	0.000898s
6	0.001764s
7	0.007941s
8	0.043s
9	0.290s
10	2.556s
11	23.976s
12	263.217s
13	3398.391s



Como é possível observar, temos aqui um algoritmo exponencial.

4. CONCLUSÃO

Concluímos respondendo a pergunta: *“Caso você fosse contratado por uma transportadora para desenvolver um sistema que encontrasse a menor rota a ser percorrida por seus caminhões, saindo e chegando do galpão de estoque e percorrendo uma única vez um conjunto de N localidades, você utilizaria a solução desenvolvida neste trabalho?”*, a resposta é não, pois como analisado durante o trabalho e de forma visível na tabela de tempo de execução com base na quantidade de entradas vemos que seria inviável utilizar isso em casos de entradas maiores que 10 entradas, pois demanda um grande espaço de tempo de execução, fazendo assim que houvesse uma grande perda de tempo, atrasando entregas.

Foi possível notar durante o desenvolvimento do trabalho prático a grande importância da Análise de Complexidade de Algoritmos ensinados pela Professora Thais Regina, pois é com ela que vemos em casos como o desenvolvido no trabalho prático que há determinados algoritmos que tornam uma solução mais otimizada, ou como visto algo inviável para a realidade.